Team 3 Final Report

SungJun Park, Ramon Arias, Yuldoshkhujaev Shakhzod, Yujin Juhn

December 2023

Abstract

SwipeBite is a user-friendly app designed to simplify restaurant selection for couples and small groups. It addresses the common dilemma of indecision when choosing where to eat, which can be time-consuming and lead to unhealthy choices. SwipeBite streamlines the decision-making process through a Tinder-style matching system, allowing users to quickly find restaurants that suit their preferences. SwipeBite focuses on providing an intuitive and visually appealing design, offering comprehensive restaurant details, including visuals, menus, pricing, parking, reviews, hours, and contact information.

This paper delves into the intricacies of developing a mobile application, particularly focusing on the implementation aspects. It aims to provide a comprehensive understanding of the internal mechanisms of the application. Furthermore, the paper will critically examine the challenges and limitations encountered during the development of SwipeBite, offering insights into the practical aspects of mobile app development.

Keywords — Restaurant Selection App, Tinder-Style Matching System, Naver Maps Crawler, Mobile App Development

1 Introduction

SwipeBite emerges as an innovative mobile application designed to revolutionize the restaurant selection process. By leveraging a user-friendly interface and a Tinder-style matching system, it aims to streamline the decision-making experience for diners, particularly focusing on couples and small groups. This application addresses the often time-consuming and indecisive process of choosing where to eat by providing an efficient, enjoyable, and informed dining choice mechanism. The development of SwipeBite not only encapsulates the challenges of modern app creation but also represents a significant step forward in combining technological advancement with everyday convenience.

1.1 Motivation

The development of SwipeBite is motivated by compelling evidence that highlights the significant time couples spend in deciding what to eat, a seemingly mundane yet crucial aspect of daily life. A survey sheds light on this issue, revealing that the average American couple dedicates approximately 132 hours each year to making decisions about their meals. This figure translates to around 21 minutes per day, a seemingly small daily interval that, however, accumulates to an astonishing total of 275 days over the course of a typical 50-year marriage. This substantial allocation of time to meal decision-making underscores a prevalent challenge faced by couples today.

Further insights from the survey indicate that the most challenging aspect for couples is deciding where to order from, with 37% of respondents citing this as a primary difficulty. This finding reflects the complexity and stress involved in choosing a dining location that satisfies both partners' preferences. In addition, the survey revealed that discussions about "what to eat for a meal" rank as the top annoying conversation among couples, with 37% finding this topic particularly bothersome.

These statistics not only highlight the time-consuming nature of meal decision-making but also suggest a level of frustration and dissatisfaction inherent in the process. The frequent occurrence of this dilemma points to a significant gap in existing solutions, one that SwipeBite aims to fill. By simplifying and streamlining the process of choosing a place to eat, SwipeBite seeks to alleviate the burden of decision-making, reduce the time spent in discussions, and potentially mitigate the annoyance associated with such conversations. Our goal is to transform meal selection from a source of contention into an enjoyable and efficient experience, thereby enhancing the overall quality of life for couples and groups.

1.2 Proposal and Goal

The importance of addressing this problem cannot be overstated. Modern society is characterized by increasingly busy schedules, and people are constantly seeking ways to optimize their time. SwipeBite aims to provide a solution by simplifying the dining decision-making process, ensuring that users can make an informed choice quickly and conveniently.

Previous attempts to tackle this issue have fallen short in several crucial aspects. Existing food-oriented apps, often touted as the "Tinder for food," suffer from multiple shortcomings.

In response to the identified shortcomings of existing dining decision-making applications, we present an innovative solution: SwipeBite, a Tinder-style app designed specifically for restaurant selection. Our approach draws inspiration from the concept of "matching" popularized by Tinder but applies it to the realm of dining choices. Instead of matching with potential dates, SwipeBite enables users to match with restaurants that align with their culinary preferences, thereby simplifying the process of choosing where to eat with significant others or friends.

Our heartfelt goal for the SwipeBite project is to:

- Provide a seamless and enjoyable solution to the perennial challenge of selecting a restaurant, particularly for couples.

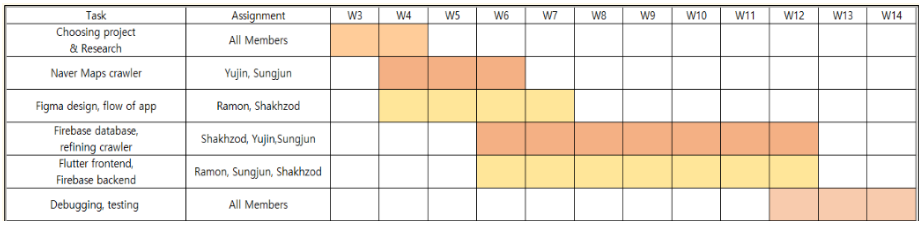
- Significantly reduce the hours spent in indecision, allowing more quality time together.
- Encourage healthier dining choices by offering comprehensive restaurant information.
- Elevate the user experience through an intuitive and aesthetically pleasing user interface.
- Extend our solution to a Korean audience while remaining sensitive to the unique needs of users from diverse cultural backgrounds.
- Simplify the dining decision-making process by introducing a matching system akin to Tinder, making the experience effortless, fun, and efficient.

1.3 Approach

SwipeBite is designed to be user-friendly and straightforward, providing users with a selection of restaurants and food establishments tailored to their preferences. The app's swiping mechanism allows users to effortlessly express their dining choices, facilitating a match when both parties show interest in the same location. This innovative approach ensures that dining decisions are made efficiently, without the need for prolonged discussions.

To enhance the user experience, we have placed a strong emphasis on developing an intuitive and visually appealing user interface. Recognizing the shortcomings of previous applications in this domain, our aim was to craft a UI that is not only aesthetically pleasing but also user-friendly and convenient, addressing the diverse needs and preferences of our users.

The development of SwipeBite followed a carefully structured timeline, ensuring that each task was systematically assigned and executed by our team. The *Figure 1* illustrates the detailed schedule of our development process, from the initial conceptualization to the final stages of debugging and testing. This timeline underscores our commitment to a methodical approach in delivering an application that simplifies the dining decision-making process and enhances user enjoyment.



Task	Assignment	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14
Choosing project & Research	All Members	🟧	🟧										
Naver Maps crawler	Yujin, Sungjun		🟧	🟧	🟧								
Figma design, flow of app	Ramon, Shakhzod		🟨	🟨	🟨	🟨							
Firebase database, refining crawler	Shakhzod, Yujin,Sungjun				🟧	🟧	🟧	🟧	🟧	🟧	🟧		
Flutter frontend, Firebase backend	Ramon, Sungjun, Shakhzod				🟨	🟨	🟨	🟨	🟨	🟨	🟨		
Debugging, testing	All Members										🟧	🟧	🟧

Figure 1: Development Timeline

1.4 Design and Implementation

In the following section, we delve into the specific design and implementation strategies that have been crafted to realize the vision of SwipeBite. From the data gathering through our Crawler to the intuitive user interface design in Figma, the efficient data management via Firebase, and the front-end development using Flutter, each element plays a role in creating a solution that is not only efficient but also enjoyable for our users.

Crawler

SwipeBite's Crawler, developed using Python and Selenium, streamlines the process of gathering restaurant information. It efficiently extracts details such as names, categories, images, addresses, hours, contact info, parking, menus, and reviews from Naver Maps, eliminating the need for users to manually search for this information.

Figma Design

The app's design, created using Figma, focuses on simplicity and user-friendliness. With an emphasis on convenience, SwipeBite features an intuitive UI that allows users to easily browse restaurants, access detailed information, and explore menus without unnecessary navigation complexities.

Firebase Database

Firebase Database is utilized for essential features like user authentication, sign-up, and room creation, as well as for storing restaurant data. It includes Firebase Authentication and Firestore Database, housing four key collections: Restaurants in Seoul, Restaurants in Suwon, Users, and Rooms, thereby ensuring efficient data management.

Flutter Implementation

SwipeBite's front-end is developed in Flutter, chosen for its ease of use. This phase involved transforming Figma designs into Flutter, enhancing the interface, and integrating essential features like room creation, filter setting, and a swiping mechanism to streamline the restaurant selection process.

1.5 Achievements

This section highlights the overarching success of the SwipeBite app, as validated by a survey conducted among a small group of users. The survey results paint a clear picture: dining out is a frequent activity for a vast majority of participants, making the app highly relevant for modern urban lifestyles. Pre-SwipeBite, users spent a considerable amount of time deciding on dining locations, a process which our app has significantly expedited. Post-adoption, most users reported a dramatic reduction in decision-making time, with many reaching a consensus in under 5 minutes. The app has not only streamlined the selection process but also achieved an impressive user satisfaction rate, indicating that SwipeBite effectively resolves the challenges of group dining decisions while enhancing user experience.

2 Design for the Proposed System

2.1 Overall Architecture

Figure 2 shows a flow of our app. Users start by signing in or joining through their Google account, then move on to set up or enter a group room. Here, they pick their dining preferences like parking availability or cuisine type. After that, each user swipes through restaurant options—right for yes, left for no—and can press an info button for more information on each spot. In the end, the app suggests a place to eat, showing pictures and other helpful details to make choosing easier.

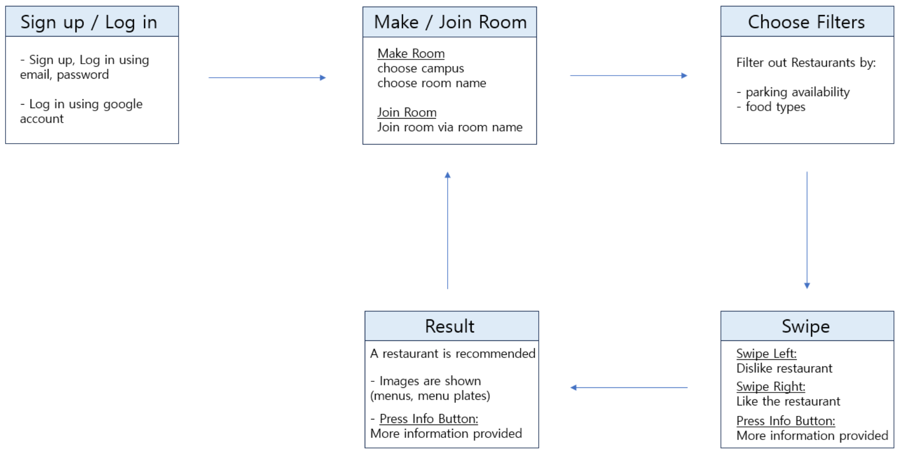


Figure 2: SwipeBite Application Flow Chart

2.2 Core Technique

The cornerstone of SwipeBite's functionality is the Tinder-style matching system. This model is instrumental in simplifying the decision-making process by allowing users to swipe through a curated selection of restaurants. This swiping mechanism is not only user-friendly but also adds an element of engagement and fun to the restaurant selection process. The design of this matchmaking system is as follows.

Gathering User Preferences

The app first retrieves information about a specific group ('room') from the database, including the restaurant preferences of each user. These preferences are stored in a way that indicates whether a user 'liked' a particular restaurant.

Counting Votes

The app tallies the 'likes' for each restaurant. Essentially, it is counting how many users in the group have expressed a preference for each restaurant.

Determining the Most Popular Choice

- If there is a restaurant that every user in the group likes (unanimous agreement), that restaurant is chosen.
- If there isn't unanimous agreement, but one restaurant has more likes than any other (a majority), then this restaurant is chosen.
- If there's no clear majority, the app randomly selects a restaurant from the list of options.

2.3 General Challenges

2.3.1 Number of Filters

One of the primary objectives of SwipeBite is to address the issue commonly faced by couples and friend groups who often spend excessive time deciding where to dine. Since our app introduces numerous dining options, it could be argued that our app might offer too many choices and only exacerbates the problem. We took various measures to ensure this was not the case. However in the process of doing that, we faced some challenges.

Key to SwipeBite's functionality is its ability to tailor restaurant options to user preferences. This personalization is achieved by prompting users to apply filters that eliminate undesired dining choices. These filters are based on two criteria: the need for parking availability and the type of cuisine preferred. During the development phase, we encountered our first significant challenge: dealing with an overwhelming number of food categories.

Upon aggregating restaurant data from Naver Maps, we were presented with 112 distinct food categories. Presenting all these categories as individual filters would be counterproductive, potentially overwhelming users with too many choices and thereby prolonging the decision process. To streamline this, we employed artificial intelligence to intelligently cluster these categories into broader groups.

For example, among the original 112 categories from Naver Maps, categories such as 감자탕 (pork back-bone stew), 고기요리 (meat dishes), 곰탕 (beef bone soup), 곱창 (grilled intestines), 국밥 (rice soup), 국수 (noodles), 꼬치 (skewers), 낙지요리 (octopus dishes), 냉면 (cold noodles), and 닭갈비 (spicy grilled chicken) were consolidated under the broader category of 한식 (Korean cuisine). This AI-driven categorization effectively reduced the original 112 categories into a more manageable set of 10 filters. This reorganization was not a removal of Naver Maps' categories but a strategic reconfiguration for simplicity and efficiency. As a result, users are now able to make quicker and more straightforward choices from a condensed list of food options.

2.3.2 Filter Logic

Upon establishing the filters to be incorporated into SwipeBite, we were confronted with a subsequent challenge: devising a method to combine individual user-selected filters within a group and then presenting a unified list of restaurant options. This task was particularly challenging as our goal was to speed up the restaurant selection process by reducing the number of choices presented to users. Simply displaying all restaurants corresponding to every filter selected by group members would result in an overwhelming array of options, contrary to our objective of simplifying the decision-making process.

To resolve this issue, we implemented a strategic approach. When there is an overlap or intersection in the filters selected by group members, the application displays only those restaurants that fit within the intersecting categories. For instance, if one user selects 한식 (Korean cuisine) and 일식 (Japanese cuisine), and another chooses 양식 (Western cuisine) and 일식, the app identifies the common category — in this case, 일식 — and consequently displays only Japanese restaurants.

However, we also had to consider scenarios where there is no intersection between the chosen filters, such as when one user selects 양식 and another 한식. In these situations, to foster a spirit of compromise and ensure that a variety of options are still available, we employed a union of the filters. This means that the application displays restaurants from both categories (양식 and 한식), thus broadening the selection while still catering to the group's diverse preferences. This solution adeptly balances the need to provide sufficient choice without overwhelming the users, thereby enhancing the overall efficiency and user experience of SwipeBite

2.3.3 Number of Restaurants

The final major challenge faced in the development of SwipeBite pertained to determining an optimal number of restaurant options to display. An excessive number of choices could potentially overwhelm users, counteracting our goal of streamlining the decision-making process.

Given the vast number of potential restaurants available under each category, the necessity of imposing a limit on the number of options presented became apparent. For instance, selecting 한식 (Korean cuisine) as a filter could yield a list of over a hundred restaurants in a given area. Presenting such a large selection to the user could lead to frustration and prolong the decision-making process.

To address this, a cap was established on the number of restaurants displayed during the matchmaking process. After careful consideration, we settled on presenting a maximum of 12 restaurants once the user applies their filters. This number was chosen as a balance between providing sufficient variety to ensure user satisfaction and avoiding an overwhelming array of choices that could lead to decision fatigue.

The decision to limit the options to 12 is not fixed and can be adjusted based on further analysis or user feedback. Although time constraints limited our ability to experiment with different thresholds, this number can be modified in future iterations of the app based on empirical evidence or user experience studies.

3 Implementation

The development of SwipeBite lied through a sequence of interdependent processes that culminated in the production of a functional application. This section outlines the essential steps of the implementation process.

3.1 Crawler Development

The initial plan for collecting restaurant data, including names, photos, and other relevant information, was to use existing APIs. Extensive study indicated that platforms like Naver Maps, Kakao Maps, and Google Maps do not support this feature. Their APIs are limited to map visualizations and do not allow for the extraction of restaurant information. As a result, we switched to web scraping as our primary data acquisition strategy, extracting information directly from Naver Maps.

Then we looked into existing scraping tools like *Octoparse* and *ParseHub*. However, these technologies were inappropriate due to cost constraints and insufficient data formatting skills. This situation led us to the decision to build our own custom web crawler.

We created a Python script that uses Selenium for web scraping in order to extract a full dataset from Naver Maps. The script begins by configuring the Chrome WebDriver to optimize the scraping activity. It then navigates the Naver Maps search interface, typing queries for restaurant listings within specified locations and interacts with the resulting web elements to obtain the search results.

The script works by going through search results page by page, and extracts information of each restaurant. It captures details such as name, category, address, operating hours, contact, parking availability and consumer reviews. In addition to this, the crawler manages to crawl images of the place itself as well as the images of the menu plate and dishes. Each restaurant's information is compiled into a detailed dictionary, and then it is saved in the form of a JSON file. This ensures accessibility and utility.

Utilizing this customized crawler, we successfully compiled the requisite datasets for the Suwon and Seoul campuses of Sungkyunkwan University.

3.2 Figma Design

Our design objective was to merge simplicity with functionality, creating an interface that is both pleasing and easy to navigate. To achieve this, we employed Figma, a widely-utilized web-based collaborative tool for interface design.

In our initial design, we focused on creating an easy-to-use interface for users to form and join groups for restaurant choices, keeping it minimal yet effective. We included features for browsing restaurants, accessing detailed information, and exploring menus. The final result of our design process is a user-friendly app that blends good looks with practicality, making it easy for users to navigate and make group decisions without the usual hassle.

3.3 Firebase Database

SwipeBite was designed with common elements like user identification, room setup, and easy data connection in mind. We chose Firebase, a platform famous for its broad portfolio of services, to support these functionalities as well as the storage of data sourced from our crawler.

Our strategy for deployment involved the use of Firebase Authentication and Firestore Database. We started by writing a Python software that could submit restaurant data in JSON format to Firestore. We then established a Firestore schema consisting of four distinct collections, namely: *Restaurants in Seoul campus and Suwon campus*, *Users* and *Rooms*.

The restaurant-related collections include documents with 12 attributes that include information such as the place's name, cuisine type, pricing, 2 address formats, visual representations of menus and dishes, venue images, parking availability, consumer reviews, contact information, and operational hours.

The 'Users' collection is designed to hold important user information such as names and email addresses. The 'Rooms' collection, on the other hand, is more complex, including options for campus selection, dietary, and parking filters. Need to mention that the filter parameter is shared collectively, with a union operation used to ensure that preferences are represented across all users. This collection also includes information about room ownership, participant details, readiness indicators, and parking filter status. This extensive data architecture powers the app's dynamic development of restaurant selections while recording and reflecting unique user preferences.

Regarding Firebase Authentication, we currently facilitate two sign-in modalities: email with password and Google authentication. Upon successful sign-in, user credentials are automatically stored within the 'Users' collection of Firestore, thus ensuring a consistent user experience.

3.4 Implementation in Flutter

Flutter, which is powered by the Dart programming language, was chosen for the application's development because to its user-friendliness and growing popularity.

The first round of front-end development required translating our preliminary Figma design into Flutter, resulting in a visual representation that corresponded to our envisioned interface. Subsequent code updates improved the aesthetics and functionality, and more pages were added to improve the user experience.

The SwipeBite application's operational logic is based on user authentication; access is required to use the room creation and participation capabilities. Following login, the main menu displays an overview of the user's active rooms, including essential statistics such as room titles and participation counts. Users have the option of joining existing rooms or starting new ones. Within each room's interface, members are listed alongside actionable features such as Filters, Ready, and Start buttons. The application is designed to activate these buttons in a sequential manner, with the preparedness condition of the room influencing the activation of future features. When the 'Start' command is issued, the system automatically generates restaurant selections based on the collective filter settings, beginning the swiping sequence. Upon the completion of this step, the application uses the swiping data to identify and recommend a mutually favored dining location. It is worth noting that the restaurant recommendations are drawn at random from the server's huge database.

This room-centric architecture is critical for keeping user-specified preferences and ensuring accurate restaurant selection. Furthermore, it simplifies the matching process by methodically preserving swipe results within Firestore, making it easier to identify common selections.

3.5 Debugging and Testing

The completion of the application's fundamental functionality was followed by an extensive debugging and testing process. This procedure uncovered a number of issues, some of which were critical to the overall customer experience. These challenges were addressed with enthusiasm. In parallel, we conducted an in-depth review of the user interface, which resulted in the redesign of several pages to improve their visual appearance. This careful phase of debugging and testing was critical in ensuring that SwipeBite not only worked smoothly but also provided an interesting and visually pleasing user experience.

4 Evaluation

In order to evaluate our application within real-world settings, small group of people were asked to use SwipeBite and then answer a survey. The study consisted of nine questions probing into various aspects such as the frequency of dining out, the duration of decision-making regarding meal selection with and without the assistance of SwipeBite, and their overall satisfaction with the application.

4.1 Dining Out Frequency

The analysis showed that a majority of the respondents, representing 57.1%, eat out multiple times within a week, while 21.4% go out once a day, which underscores the potential daily impact of SwipeBite. This trend is particularly pronounced in urban environments where dining out is a common occurrence.

4.2 Decision Time Analysis

Before the usage of SwipeBite, approximately 64.3% of the study's participants have spent an average of fifteen minutes when selecting a place to eat with their friends or significant other, indicating that it is often difficult to come to a consensus regarding a food to eat.

With SwipeBite, a marked improvement was observed, with 50% of users able to make a decision in roughly five minutes, and an additional 35.7% doing so in even less time. These findings affirm the app's effectiveness in enhancing the efficiency of the decision-making process.

4.3 User Experience Insights

User satisfaction is a critical determinant of an application's success. In our evaluation, 28.6% of participants awarded SwipeBite the perfect score of 5 out of 5, while a predominant 50% rated it 4 out of 5. According to the findings of this study, the app does a decent job of assisting individuals in deciding where to eat, which reflects positively on both its utility and user experience.

4.4 Feature Feedback

Participants provided valuable feedback on SwipeBite's features. The restaurant database, collected by crawling the Naver Maps, was highly praised, with 42.9% favourability, reflecting the value users place on the quality and diversity of options. Room creation was also well-received with 35.7% of users choosing this option. While the swiping function is essential, 21.4% of users saw it as an area with room for improvement, implying the need for ongoing enhancements.

In terms of development areas, the study highlighted UI/UX as the primary contender, as indicated by 35.7% of the users. Such feedback is useful in directing our efforts toward creating a more intuitive and engaging user experience. The Dataset was also reported for improvement by 21.4% percent of users, indicating a potential to expand our restaurant library. Furthermore, 14.3% percent of users mentioned swiping and filters as functional but improveable features, emphasizing the importance of incremental improvement.These assessments are crucial to our continuous improvement strategy, as they ensure that SwipeBite not only meets, but exceeds, user expectations in future revisions. Our goal continues to be to fine-tune the application's fundamental functions, resulting in an incomparable, enjoyable, and efficient platform for dining decision-making.

4.5 Adoption and Recommendation Potential

The survey data on future SwipeBite engagement and willingness to promote the program to peers produced positive results. A sizable majority, 57.1% percent, of respondents rated it '5', indicating a strong desire to incorporate SwipeBite into their future eating experiences. In parallel, 28.6% gave it a '4', and 14.3% gave it a '3', showing a good attitude regarding ongoing application use with little disagreement.

Similarly, 57.1% of participants gave SwipeBite the highest recommendation score. A sizable 35.7% indicated a strong chance to recommend, while just 7.1% provided a moderate endorsement grade. The lack of negative ratings in the lower spectrum ('1' or '2') emphasizes the minimal hesitation to support SwipeBite's adoption.

Overall, these assessments highlight SwipeBite's prospective trajectory toward becoming an essential tool in the digital eating landscape, thanks to its user-centric design and effective problem-solving skills.

5 Limitations and Discussions

5.1 Assumptions

First, we assume the information from Naver is accurate. Currently, our database relies heavily on Naver Maps since we get all of our restaurant data from there using our crawler. This means that if the information on Naver is incorrect, our display would also be inaccurate. However, we believe we can address this issue by creating an administrator email to receive corrections of incorrect information from users. Furthermore, to encourage more people to participate, we can consider introducing a rewards system for achieving certain 'milestones' or regularly hosting prize draws for those who report issues, offering gifts as incentives.

As mentioned in our 'Implementation' section, after users select their filters, we utilize the union of these filters to retrieve restaurants from the database. We consider the union approach to be superior because it encourages compromise and opens up opportunities for users to discover new cuisines or restaurants they might not have considered individually. Often, users who initially express dislike for certain options may change their minds upon encountering unexpected or unfamiliar restaurants(menus). Therefore, we operate under the assumption that selecting the union of filters chosen by users in a group yields better results than opting for the intersection of those filters.

In our app, users are required to join rooms to swipe for restaurant selections. We have designed each room to have a unique ID, but this approach presents two main challenges. Firstly, there is the risk that users might accidentally enter the wrong room. Secondly, it can be a burden for users to continually come up with new, unique room names, which could be a frustrating experience. To address these issues, we are considering two solutions. The first solution is to implement expiring dates for rooms. This would mean that room IDs are only temporary, reducing the likelihood of confusion and overlap. The second solution involves making a tuple of the room name and the owner's ID a unique key. This would ensure that each room can be uniquely identified by its name within the context of its creator, easing the process of room creation and identification for users.

5.2 Limits and Constraints

The development of SwipeBite was subject to several limitations and constraints that influenced the scope and testing of our app. These constraints stem from the time frame, platform choices, and resource availability:

The development of SwipeBite was constrained by a tight schedule, as we had only one semester's duration to design, develop, and test the app comprehensively. This constrained our ability to thoroughly test the app with a wide range of use cases and diverse user groups.

Due to the time constraints mentioned above, we were compelled to make choices regarding the features to include in SwipeBite. The limited development period meant that each new feature addition required a significant time investment. Consequently, we focused on prioritizing essential features to ensure a functional and reliable core application.

SwipeBite was primarily developed and tested on the Android platform. While our frontend implementation utilized Flutter, which has the capability to deploy on both iOS and Android with the same codebase, thorough testing on iOS presented a challenge. This challenge arose from the need for Apple's integrated development environment (IDE), Xcode, which is exclusive to Mac computers. During our development period, we did not have convenient access to Mac computers, further limiting our ability to perform comprehensive testing on iOS.

6 Related Works

Of course, the concept of a Tinder-like application applied to food is not new, and there have been other projects with similar features, as we have discussed. However, most of them have several limitations that we aim to address in SwipeBite. In this section, we will discuss three main predecessors of our idea and outline the constraints of each of these works.

- **Food Match - Find Where to Eat**
  This application is one of the most popular products in this field. It boasts

a user-friendly design, several filters for distance and food type, and it follows the same logic as SwipeBite. Nevertheless, this application lacks a menu board, which is important for users. It also lacks information about the working hours of the restaurant, making it difficult for users to know if the matched place is currently open. The most critical limitation is that this project was originally designed for users in the United States and does not have much information about Korea and its restaurants. This limitation leads to another issue of not having a language selection option. In summary, while the Food Match application aligns with SwipeBite's concept, it has several limitations and drawbacks, making it less useful for the Korean community.

- **Entree**
  Another application that employs our Tinder-like concept can be found in Entree. Similar to the previous application, it allows users to set filters for cuisine and even by meat type. Additionally, it includes a separate section called "Recommended for you," which suggests food based on users' preferences over time. However, despite being first proposed in 2016 and initially available regionally in Los Angeles, New York, and Washington D.C., Entree is still in beta testing as of now. This suggests that the project will take some time before its full release. Moreover, even upon full release, it is likely that Entree will have the same limitation of not being designed for Korea, resulting in a limited number of restaurants and no language selection option. Furthermore, its user interface does not align with modern design principles and appears unappealing and outdated. In summary, while Entree appears ambitious in its field, with features such as food recommendations, it may not gain much popularity in Korea due to its limitations and issues.

- **Dining Code**
  This is perhaps the only application we have found that can assist users in choosing where to eat while being in Korean. It features an appealing design with a variety of useful filters. Using this app, users can search for famous restaurants in Korea based on food cuisine. They can also change their location to find the nearest options. However, Dining Code differs from SwipeBite's concept in terms of food searching. Dining Code primarily serves as a map for famous restaurants, offering various places to eat without helping users decide what to eat. This can lead to the time-consuming dilemma of food indecision. In SwipeBite, we aim to address this issue and provide assistance in selecting food. In summary, the Dining Code application is very useful for having a vast data set of restaurants per location in Korea, but it falls short in helping users choose food.

The previous approaches for finding food are not inherently bad, and some of them are quite useful. Nevertheless, each of the applications listed above has several significant drawbacks, making them unsuitable for solving the issue we originally intended to address: the problem of saving time while deciding what to eat in Korea.

7 Conclusion

We successfully implemented our initial semester goal by creating a unique service that offers personalized suggestions based on various filters set either by groups of friends or individual users.The app is intuitive and fun to use, with an attractive and user-friendly design that adds to its appeal.Ultimately, our project successfully resolved the often time-consuming and stressful task of deciding what to eat.

Throughout the project, we effectively applied theoretical knowledge gained from our academic courses to real-world challenges. A notable instance was utilizing our learnings from the "Introduction to Database" class, which enabled us to detect and resolve a race condition issue, a critical aspect for the functionality of our application.

Moreover, our team's commitment and the project's practicality were validated through actual use. Team members have already used the app to decide on meal options, exemplifying its utility and relevance. One member mentioned plans to use it with their roommates to determine daily meal choices, further demonstrating the app's potential in facilitating group decision-making in everyday life. This real-life application and positive feedback from our peers highlight the successful achievement of our course objectives and the practical impact of our project.

References

[1] Amy James, *Couples Spend A LOT Of Time Every Year Deciding What To Eat — iHeartRadio.* (n.d.). Retrieved December 10, 2023, from `https://news.iheart.com/featured/amy-james/content/2017-12-12-couples-spend-a-lot-of-time-every-year-deciding-what-to-eat/`

[2] DigitalHubUSA, *What's for dinner? Average couple argues this many times a year about what to order.* (2020, October 28). Digitalhub US. `https://swnsdigital.com/us/2020/10/whats-for-dinner-average-couple-argues-this-many-times-a-year-about-what-to-order/`

[3] N. Y. Post, *Couples Spend 5.5 Days A Year Deciding What To Eat (Video).* (2017, November 19). South Florida Reporter. `https://southfloridareporter.com/couples-spend-5-5-days-year-deciding-eat-video/`